Basic calculations with *Mathematica*

Reminder

We recall a few basic concepts from the previous chapter.

Mathematica consists of two parts: the Kernel (that actually does the calculations) and the Front End (the "windows" application that one uses to enter notation and send and retrieve the calculations from the Kernel).

The usual way of starting *Mathematica* in a Windows platform is: click "Start", point to "All Programs", and choose "*Mathematica* 6.x" from the *Mathematica* 6.x program group.

In order to access the help browser, go to "Help" and select "Documentation Center...". Use the text field to enter search terms. Click "Go" to display the associated notebook. The highlighted items in the columns track your search path. The resulting help notebook is displayed below the columns. To get information on a built-in function: with the cursor immediately after a function name, press F1; the Help Browser opens and displays the information from "Built-in functions".

Your first *Mathematica* calculations

You can use *Mathematica* just like a calculator. Type your input, press SHIFT-RET or ENTER, and *Mathematica* returns the answer.

The first time you run a calculation with *Mathematica*, you will have to wait for a few seconds until the Kernel is loaded. The Kernel will run while the Front End is open or until you explicitly close it down. When you exit the Front End, the Kernel is automatically closed.

Basic arithmetic

We are about to start our first Mathematica calculations.

Type 3+5 and execute the command (by pressing SHIFT-RET or ENTER).

```
In[137]:= 3 + 5

Out[137]= 8
```

You have done your first *Mathematica* calculation. It is easy to get familiar with the basic arithmetic operations.

Each of these represents multiplication: **`a*b a SPACE b a(b+1); 2x`** means **`2*x`**. These are standard arithmetic operations: **`2+3 2-3 2/3 2^3`**.

```
In[138]:= (2^3 - 1) / 3
```

$$\frac{7}{3}$$

Out[138]=

The precedence rules for the operations are the usual ones. You have to use brackets to change them. Note that exponentiation goes from right to left whereas the other operations go from left to right.

```
In[139]:= (12 / 3) / 4

Out[139]= 1

In[140]:= 12 / (3 / 4)

Out[140]= 16

In[141]:= 12 / 3 / 4

Out[141]= 1

In[142]:= (2^6)^3

Out[142]= 262 144

In[143]:= 2^(6^3)

Out[143]= 105 312 291 668 557 186 697 918 027 683 670 432 318 895 095 400 549 111 254 310 977 536

In[144]:= 2^6^3

Out[144]= 105 312 291 668 557 186 697 918 027 683 670 432 318 895 095 400 549 111 254 310 977 536
```

More examples:

In[145]:=
```
((2 + 1) / 4)^2
```
Out[145]= $\frac{9}{16}$

In[146]:=
```
(2 + 1) / 4^2
```
Out[146]= $\frac{3}{16}$

In[147]:=
```
2 + 1 / 4^2
```
Out[147]= $\frac{33}{16}$

In[148]:=
```
(2 + 1 / 4)^2
```
Out[148]= $\frac{81}{16}$

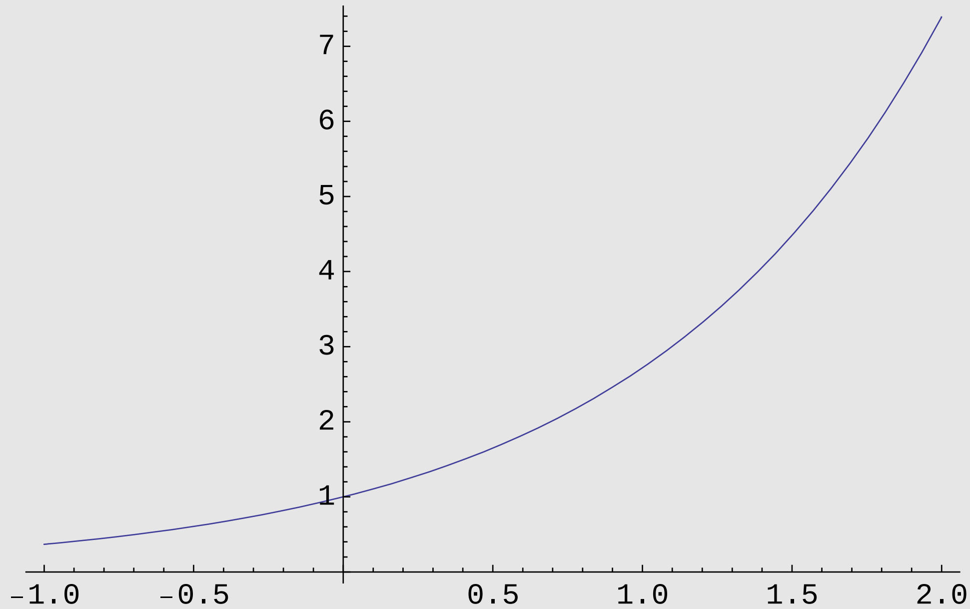
Note how *Mathematica* gives the result as a fraction, in the usual mathematical notation.

Exponential, logarithm and trigonometric functions

The exponential function e^x is represented as `Exp[x]`. The logarithm $\log(x)$ is represented by `Log[x]` (by log we understand the "natural" logarithm where the Euler number e is the base). We anticipate the plotting function `Plot`.

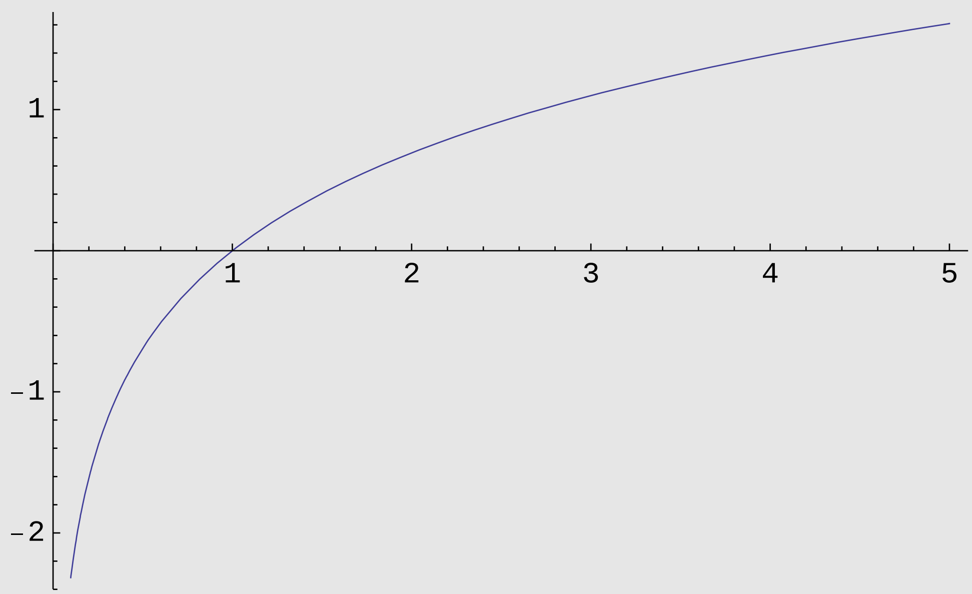
In[149]:=
```
Plot[Exp[x], {x, -1, 2}]
```

Out[149]=



In[150]:=
```
Plot[Log[x], {x, 0, 5}]
```

Out[150]=



The following table summarises the trigonometric functions, their inverses and their hyperbolic counterparts. Note that arguments are given in radians. A radian is the angle of the arc of circumference whose length equals the radius of that circumference; hence, 2π radians are 360°. (Strictly speaking, these functions are defined for certain open sets of the complex numbers $\mathbb{C}$.)

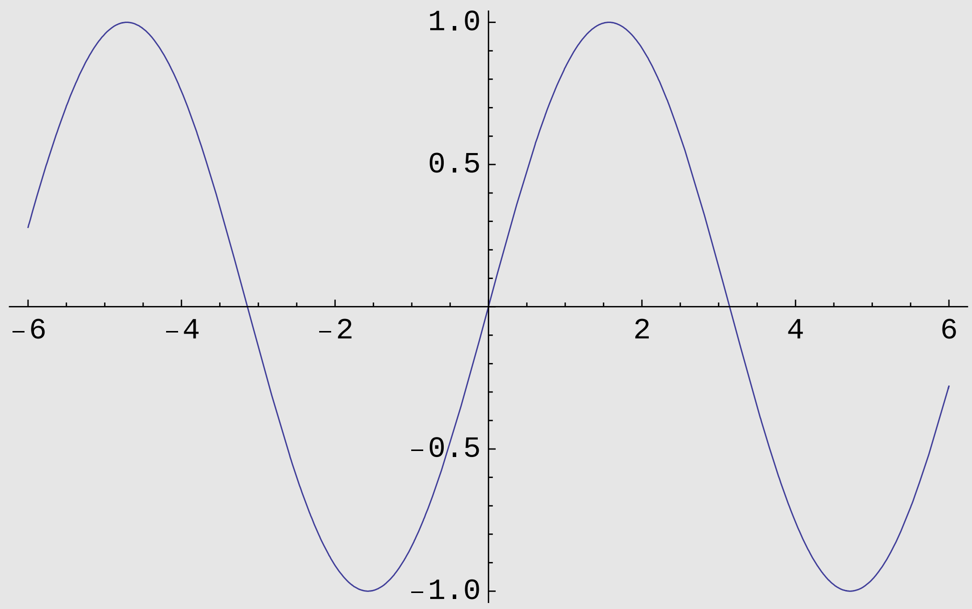
`Sin[z], Cos[z], Tan[z], Csc[z], Sec[z], Cot[z]`	trigonometric
`ArcSin[z], ArcCos[z], ArcTan[z], ArcCsc[z], ArcSec[z], ArcCot[z]`	inverse trigon
`ArcTan[x, y]`	the argument
`Sinh[z], Cosh[z], Tanh[z], Csch[z], Sech[z], Coth[z]`	hyperbolic fu
`ArcSinh[z], ArcCosh[z], ArcTanh[z], ArcCsch[z], ArcSech[z], ArcCoth[z]`	inverse hyperl

It is worthwhile to mention here that *all* Mathematica *built-in functions have their first character capitalised*. Moreover, if a function takes parameters (such as the previous ones), then these parameters are enclosed by square brackets. Moreover, *Mathematica* makes distinction between normal and capital letters.

You can try plotting these functions. For that, use the command `Plot` as before. For example:

In[151]:=
```
Plot[Sin[x], {x, -6, 6}]
```

Out[151]=



Important constants

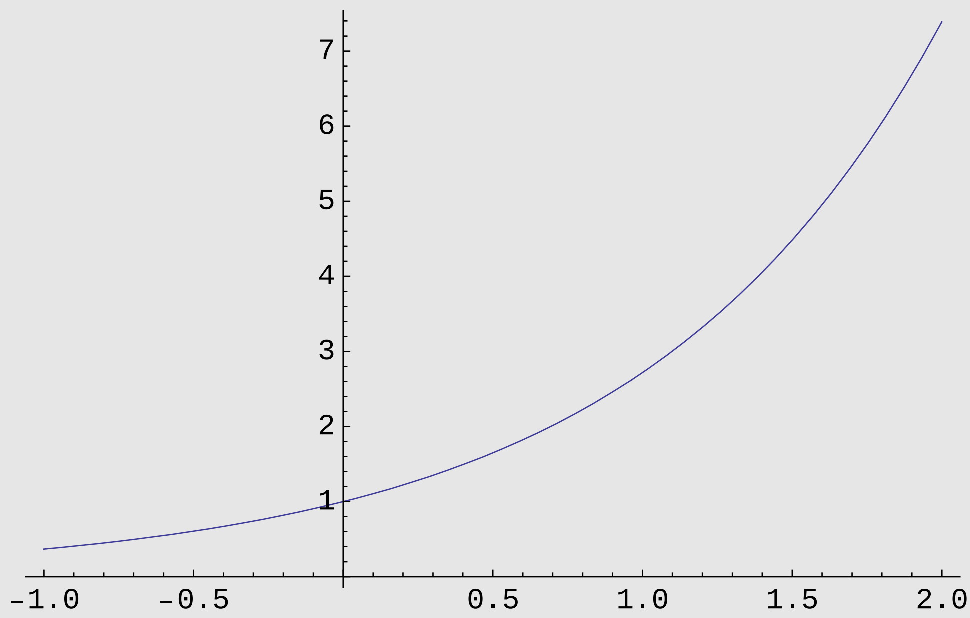
The following table summarises important constants often used in mathematics.

`Pi`	$\pi \simeq 3.14159$
`E`	$e \simeq 2.71828$ (normally output as e)
`I`	$i = \sqrt{-1}$ (normally output as i)
`Infinity`	∞

The Euler number e is represented by **`E`**:

In[152]:=
```
Plot[E^x, {x, -1, 2}]
```

Out[152]=



It is better to use the function `Exp` rather than `E^x` for efficiency issues.

We can combine functions, constants and usual arithmetic operations.

In[153]:=
```
(3 Sin[Pi / 4] + 1) ^2
```

Out[153]=

$$\left(1+\frac{3}{\sqrt{2}}\right)^2$$

Unless you explicitly say so, *Mathematica* always tries to give an *exact* result. In the following section we will see how to get approximate results. For the moment keep in mind the fact that you will get exact numbers whenever it is possible.

For example, with a normal calculator you would get an approximation of the following calculation. The precision you would obtain would depend on how powerful your calculator is. In *Mathematica*, however, you get the full integer number:

In[154]:=
```
2^1000
```

Out[154]=
```
10 715 086 071 862 673 209 484 250 490 600 018 105 614 048 117 055 336 074 437 503 \
 883 703 510 511 249 361 224 931 983 788 156 958 581 275 946 729 175 531 468 251 871 \
 452 856 923 140 435 984 577 574 698 574 803 934 567 774 824 230 985 421 074 605 062 \
 371 141 877 954 182 153 046 474 983 581 941 267 398 767 559 165 543 946 077 062 914 \
 571 196 477 686 542 167 660 429 831 652 624 386 837 205 668 069 376
```

You also get exact results with large denominators.

In[155]:=
```
195 345 134 / 3 423 423 523 + 324 213 423 / 99 569 118
```

Out[155]=
$$\frac{376\,456\,733\,822\,840\,347}{113\,622\,420\,241\,854\,238}$$

The imaginary unit is represented by **`I`**. You can perform calculations with complex numbers as well (and again you get exact results by default).

In[156]:=
```
E^(-I Pi)
```

Out[156]=
```
-1
```

In[157]:=
```
(3 + 4 I) + (-2 + I)
```

Out[157]=
```
1 + 5 ⅈ
```

In[158]:=
```
(3 + 4 I) (-2 + I)
```

Out[158]=
```
-10 - 5 ⅈ
```

Other functions

Mathematica includes a very large collection of mathematical functions. See Section 3.2 of *The Mathematica Book* for a complete list. Apart from the previous ones, the following are also useful

`Sqrt[`*x*`]`	square root ($\sqrt{x}$)
n`!`	factorial (product of integers 1, 2, …, *n*)
`Abs[`*x*`]`	absolute value
`Mod[`*n*`,` *m*`]`	*n* modulo *m* (remainder on division of *n* by *m*)
`Random[ ]`	pseudorandom number between 0 and 1
`Max[`*x*`,` *y*`, … ], Min[`*x*`,` *y*`, … ]`	maximum, minimum of *x*, *y*, …

Stopping calculations

Although *Mathematica* is a powerful calculating tool, it has its limits. Sometimes it will happen that the calculations you tell *Mathematica* to do are too complicated, or maybe the output produced is too long. In these cases, *Mathematica* could be calculating for too long to get an output so you might want to stop these calculations. It is important to know how to do this.

To abort a calculation: go to "Kernel" and select "Abort evaluation" or press ALT-[.].

It can take long to abort a calculation. If the computer does not respond an alternative is to close down the Kernel. By doing this you do not lose the data displayed in your notebooks but you do lose all the results obtained so far from the Kernel, so in case your are running a series of calculations, you would have to start again.

To close down the Kernel: go to "Kernel" and select "Quit Kernel" and then "Local".

Closing down the Kernel is not a practise that is done only when a you want to stop a calculation. Sometimes, when you have been using *Mathematica* for a long time you forget about the definitions and calculations that you have done before (maybe you have defined values for variables or functions, for example). Those definitions can clash with the calculations you are doing, so you might want to close down the Kernel and start your new calculations from scratch. In general, it is a good idea to close down the Kernel after you have finished with a series of calculations, so that when you move to a different problem your new calculations do not interact with the previous ones.

Floating point calculations

Mathematica can perform not only calculations with exact numbers but also with floating point numbers, and even arbitrary precision numbers. The arithmetic of arbitrary precision numbers is rather complicated (because numbers with different precision can be mixed) and we will not cover this topic in full in this course. We content ourselves with a few basic remarks.

expr`//N` or `N[`*expr*`]`	approximate double–precision numerical value of *expr*
`N[`*expr*`,` *n*`]`	numerical value of *expr* calculated with *n*-digit precision

To get a double-precision approximation of an exact number use `N`. This is an approximation of a previous result:

In[159]:=
```
N[(3 Sin[Pi / 4] + 1)^2]
```

Out[159]=
```
9.74264
```

Floating point numbers are represented in the usual way, and by default double-precision is assumed.

In[160]:=
```
1.5 + 2.9 / 0.1
```

Out[160]=
```
30.5
```

If you mix exact numbers with floating point numbers you get a floating point number (although I do not recommend doing these things).

In[161]:=
```
1 / 2 + 0.5
```

Out[161]=
```
1.
```

You can get any precision you want from an *exact* number. To achieve that, one specifies the desired precision as the second argument of `N`.

In[162]:=
```
N[(3 Sin[Pi / 4] + 1)^2, 50]
```

Out[162]=
```
9.7426406871192851464050661726290942357090156261308
```

In[163]:=
```
N[Pi, 200]
```

Out[163]=
```
3.14159265358979323846264338327950288419716939937510582097494459230781\
 6406286208998628034825342117067982148086513282306647093844609550582231\
 7253594081284811174502841027019385211055596446229489549303820
```

Bear in mind that no matter the precision you give to a number, your result is always an approximation and therefore it is subject to round-off error. It is not a bad idea to perform all calculations with exact numbers and only get the approximation at the end. This is not always possible, for example if you are programming numerical methods to get approximate solutions of problems that cannot be solved exactly. In that case, you might want to start with double-precision numbers from scratch (it is not very good to mix exact numbers with floating-point numbers). Latter, if you need more precision in your calculations you should modify your previous algorithms to arbitrary precision.

A final remark is that you cannot increase the precision of a number with `N`. For example, in the following calculation you cannot get 50 digits of precision because the result has double-precision (usually 16 digits, although *Mathematica* does not display them all).

In[164]:=
```
N[Sin[0.5], 50]
```

Out[164]=
```
0.479426
```

Here, the result is exact, so we can get the precision we want:

In[165]:=
```
N[Sin[1 / 2], 50]
```

Out[165]=
```
0.47942553860420300027328793521557138808180336794060
```

The following line specifies that 0.5 has *already* 50 digits of precision and hence the final result has more precision than usual. However, this is a more advanced topic and we will not go through it.

In[166]:=
```
Sin[0.5`50]
```

Out[166]=
```
0.47942553860420300027328793521557138808180336794060
```